

BCA611 Video Oyunları için 3B Grafik

WebGL'e giriş

Zümra Kavafoğlu

WebGL

- **WebGL (Web Graphics Library)** 3B grafiklerin, uyumlu web tarayıcılarında herhangi eklentiye(plug-in) ihtiyaç olmadan renderlanmasını sağlayan bir JavaScript API'sidir.
- Kullanımı OpenGL'e çok benzerdir.
- Herhangi bir kurulum gerektirmez.
- Sistem bağımlı değildir.

<http://learningwebgl.com>

İlk WebGL programı: Canvas oluşturma ve renklendirme

```
1  <!doctype html>
2  <html>
3  <head>
4    <title>Background Color</title>
5    <style>
6      body{ background-color: grey; }
7      canvas{ background-color: white; }
8    </style>
9    <script>
10     var gl = null,
11         canvas = null;
12
13     function initWebGL()
14     {
15       canvas = document.getElementById("my-canvas");
16       try{
17         gl = canvas.getContext("webgl");
18       }catch(e){
19       }
20
21       if(gl)
22       {
23         setupWebGL();
24       }else{
25         alert( "Error: Your browser does not appear to support WebGL.");
26       }
27     }
28
29     function setupWebGL()
30     {
31       //set the clear color to a shade of green
32       gl.clearColor(0.08, 0.74, 0.8, 1.0);
33       gl.clear(gl.COLOR_BUFFER_BIT);
34
35       gl.viewport(0, 0, canvas.width, canvas.height);
36     }
37
38
39   </script>
40 </head>
41 <body onload="initWebGL()">
42   <canvas id="my-canvas" width="800" height="600">
43     Your browser does not support the HTML5 canvas element.
44   </canvas>
45 </body>
46 </html>
```

HTML <canvas> elemanı bir web sayfasında grafik çizmek için kullanılan alandır.

İlk WebGL programı: Canvas oluşturma ve renklendirme

```
1  <!doctype html>
2  <html>
3  <head>
4    <title>Background Color</title>
5    <style>
6      body{ background-color: grey; }
7      canvas{ background-color: white; }
8    </style>
9    <script>
10     var gl = null,
11         canvas = null;
12
13     function initWebGL()
14     {
15       canvas = document.getElementById("my-canvas");
16       try{
17         gl = canvas.getContext("webgl");
18       }catch(e){
19       }
20
21       if(gl)
22       {
23         setupWebGL();
24       }else{
25         alert( "Error: Your browser does not appear to support WebGL.");
26       }
27     }
28
```

Tarayıcı webgl destekliyorsa null'dan farklı bir değer döner

İlk WebGL programı: Canvas oluşturma ve renklendirme

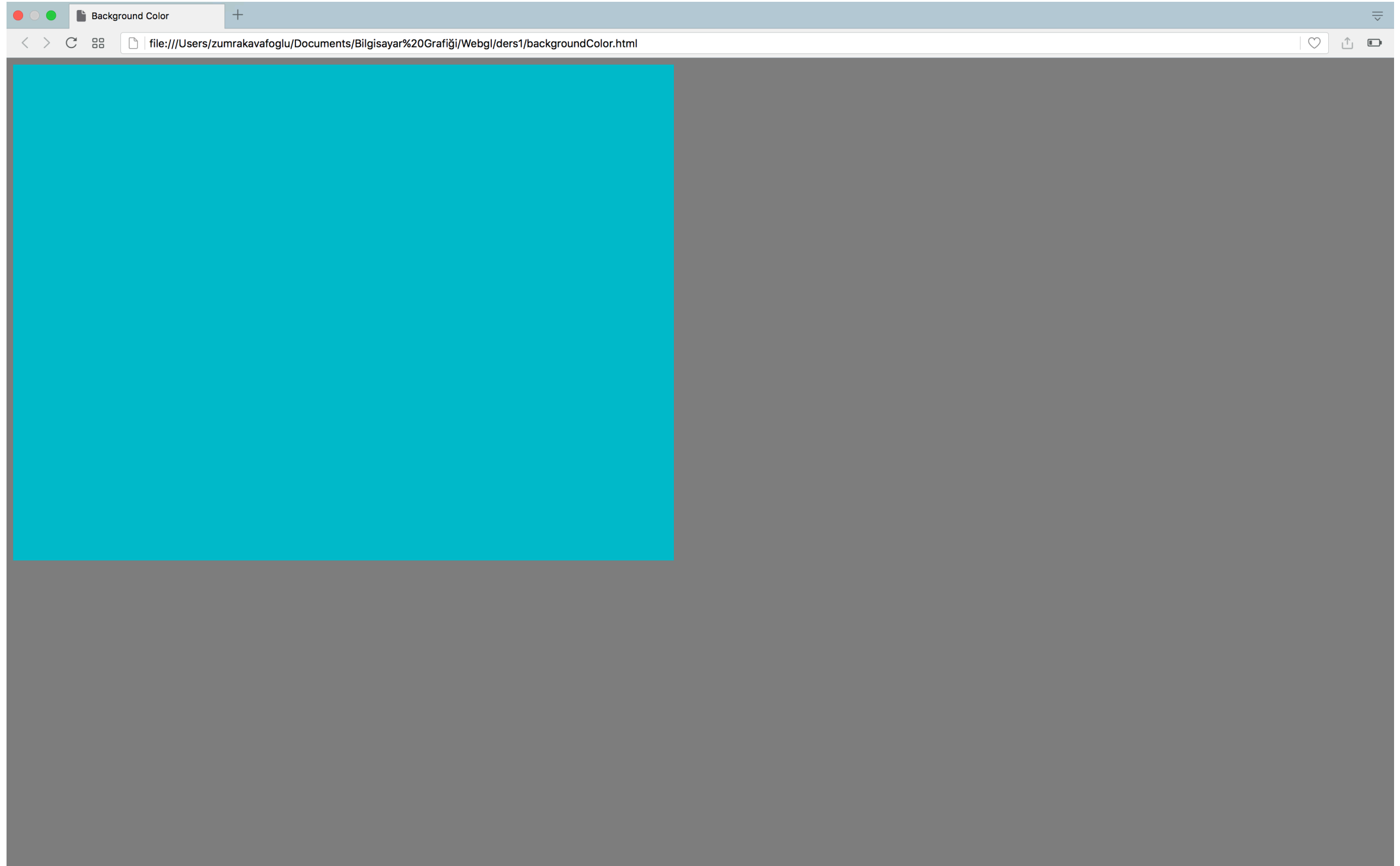
```
29 function setupWebGL()
30 {
31     //set the clear color to a shade of green
32     gl.clearColor(0.08, 0.74, 0.8, 1.0);
33     gl.clear(gl.COLOR_BUFFER_BIT);
34
35     gl.viewport(0, 0, canvas.width, canvas.height);
36 }
37
38
39 </script>
40 </head>
41 <body onload="initWebGL()">
42     <canvas id="my-canvas" width="800" height="600">
43     Your browser does not support the HTML5 canvas element.
44     </canvas>
45 </body>
46 </html>
```

*Arkaplan rengini
değiştirir.*

*Çizim yapılacak
alanın boyutlarını
belirler*

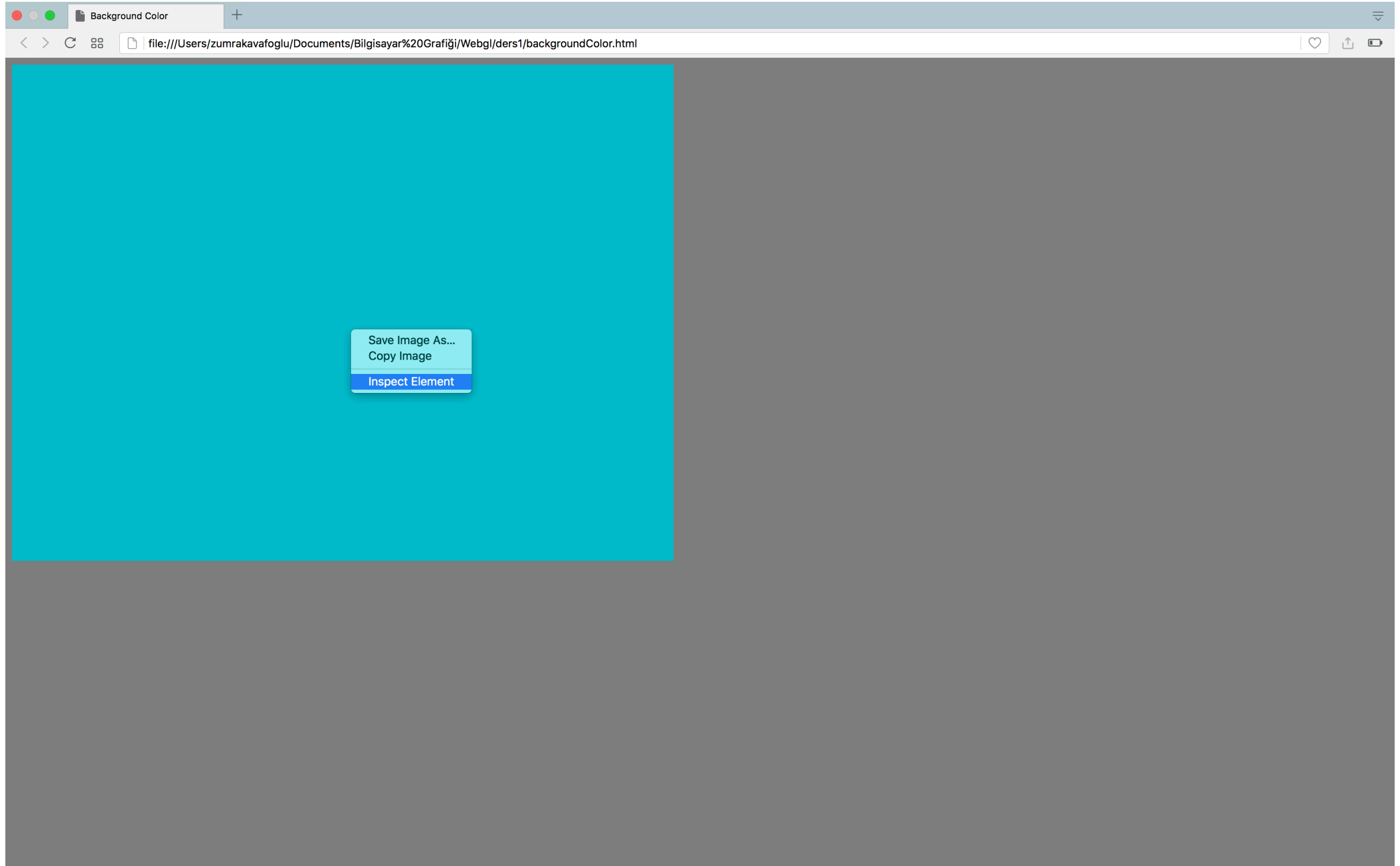
İlk WebGL programı: Canvas oluşturma ve renklendirme

Dosyayı .html uzantısıyla kaydedip çalıştırdığımızda tarayıcımızda bu sayfa açılır.



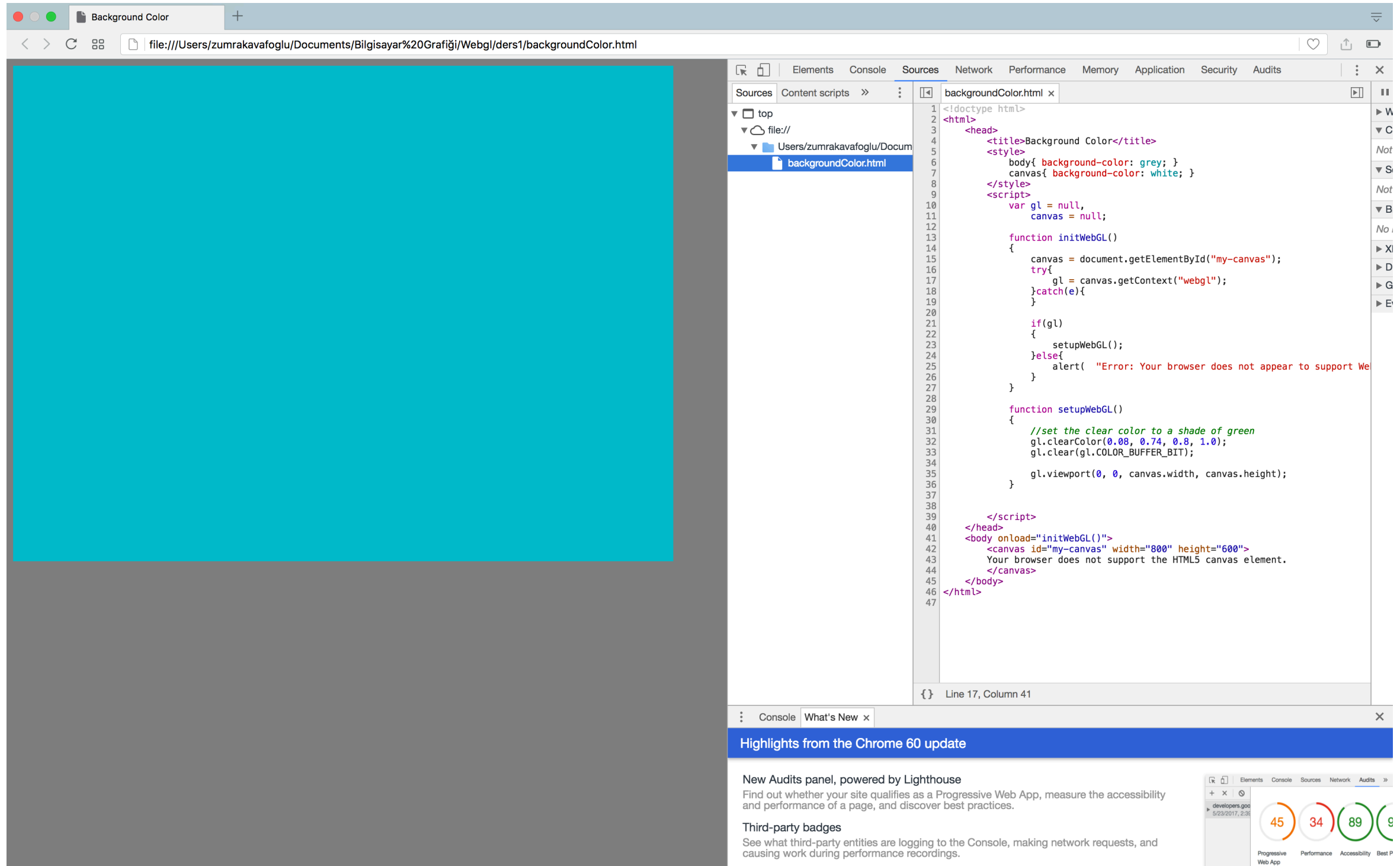
İlk WebGL programı: Canvas oluşturma ve renklendirme

Bu sayfa üzerinde kodumuzu ve varsa hatalarını görebiliriz.



İlk WebGL programı: Canvas oluşturma ve renklendirme

Bu sayfa üzerinde kodumuzu ve varsa hatalarını görebiliriz.



The screenshot shows a web browser window with a single tab titled "Background Color". The address bar shows the file path: `file:///Users/zumrakavafoglu/Documents/Bilgisayar%20Grafik%20Webgl/ders1/backgroundColor.html`. The main content area displays a solid cyan rectangle, which is the WebGL canvas. The Chrome DevTools interface is open, showing the "Sources" panel. The file `backgroundColor.html` is selected, and the code is visible. The code is a WebGL program that initializes a canvas and sets a clear color. The code is as follows:

```
1 <!doctype html>
2 <html>
3   <head>
4     <title>Background Color</title>
5     <style>
6       body{ background-color: grey; }
7       canvas{ background-color: white; }
8     </style>
9     <script>
10      var gl = null,
11          canvas = null;
12
13      function initWebGL()
14      {
15        canvas = document.getElementById("my-canvas");
16        try{
17          gl = canvas.getContext("webgl");
18        }catch(e){
19        }
20
21        if(gl)
22        {
23          setupWebGL();
24        }else{
25          alert( "Error: Your browser does not appear to support WebG
26        }
27      }
28
29      function setupWebGL()
30      {
31        //set the clear color to a shade of green
32        gl.clearColor(0.08, 0.74, 0.8, 1.0);
33        gl.clear(gl.COLOR_BUFFER_BIT);
34
35        gl.viewport(0, 0, canvas.width, canvas.height);
36      }
37
38    </script>
39  </head>
40  <body onload="initWebGL()">
41    <canvas id="my-canvas" width="800" height="600">
42      Your browser does not support the HTML5 canvas element.
43    </canvas>
44  </body>
45 </html>
```

The code is a WebGL program that initializes a canvas and sets a clear color. The code is as follows:

Line 17, Column 41

Console | What's New x

Highlights from the Chrome 60 update

New Audits panel, powered by Lighthouse

Find out whether your site qualifies as a Progressive Web App, measure the accessibility and performance of a page, and discover best practices.

Third-party badges

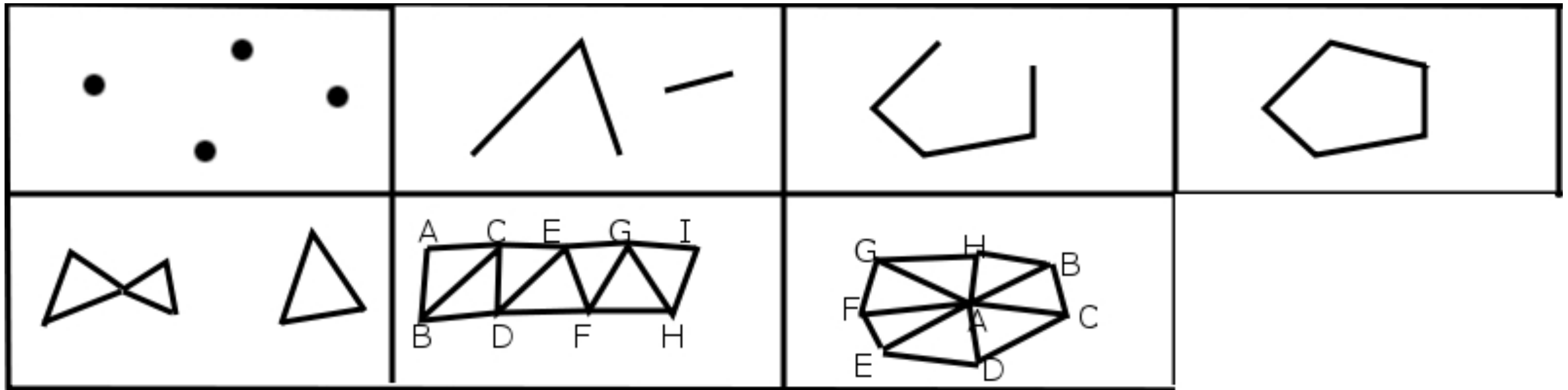
See what third-party entities are logging to the Console, making network requests, and causing work during performance recordings.

Progressive Web App | Performance | Accessibility | Best P

45 | 34 | 89 | 9

WebGL primitif tipleri

Primitifler bir grafik API'siyle oluşturulan tüm modellerin yapıtaşlarını oluştururlar.
Primitiflerin yapıtaşları ise köşelerdir.



Üst satır: POINTS, LINES, LINE_STRIP, LINE_LOOP

Alt satır: TRIANGLES, TRIANGLE_STRIP, TRIANGLE_FAN

OpenGL'in aksine WebGL QUAD primitifine sahip değildir.

VBO (Vertex Buffer Object / Köşe Belleği Nesnesi)

VBO, grafik kartında yer alan ve köşeler hakkında bilgi tutmak için tasarlanmış yüksek hızlı bir geçici bellektir(memory buffer)

Bir köşe ile ilişkili tüm veri JavaScript API'sinden GPU'ya aktarılmalıdır. WebGL'de köşenin özelliklerini tutan VBO'lar oluşturulmalıdır. Bu VBO'lar daha sonra köşe ile ilgili veriyi işleyen shader programlarına gönderirler. Shaderlarla ilgili detaylı bilgiyi ileriki derslerde göreceğiz.

Her VBO, köşenin bir özelliğini tutar, bu özellik pozisyon, renk, normal vektörü, doku koordinatları vs. olabilir.

Tek bir üçgenin köşe pozisyonları için VBO oluşturma

(-0.5,-0.5) , (0.5,-0.5) ve (0,0) köşe pozisyonlarına sahip bir üçgen çizdirmek için bu pozisyonları tutan bir VBO oluşturmamız.

```
function setupBuffers()
{
    var triangleVertices = [
        -0.5, -0.5, 0.0,
        0.5, -0.5, 0.0,
        0.0, 0.0, 0.0,
    ];

    triangleVertexPositionBuffer = gl.createBuffer();
    gl.bindBuffer(gl.ARRAY_BUFFER, triangleVertexPositionBuffer);
    gl.bufferData(gl.ARRAY_BUFFER, new Float32Array(triangleVertices), gl.STATIC_DRAW);
}
```

Tek bir üçgenin köşe pozisyonları için VBO oluşturma

(-0.5,-0.5) , (0.5,-0.5) ve (0,0) köşe pozisyonlarına sahip bir üçgen çizdirmek için bu pozisyonları tutan bir VBO oluşturmamız.

```
function setupBuffers()
{
    var triangleVertices = [
        -0.5, -0.5, 0.0,
        0.5, -0.5, 0.0,
        0.0, 0.0, 0.0,
    ];

    triangleVertexPositionBuffer = gl.createBuffer();
    gl.bindBuffer(gl.ARRAY_BUFFER, triangleVertexPositionBuffer);
    gl.bufferData(gl.ARRAY_BUFFER, new Float32Array(triangleVertices), gl.STATIC_DRAW);
}
```



Buffer oluştur

Tek bir üçgenin köşe pozisyonları için VBO oluşturma

(-0.5,-0.5) , (0.5,-0.5) ve (0,0) köşe pozisyonlarına sahip bir üçgen çizdirmek için bu pozisyonları tutan bir VBO oluşturmamız.

```
function setupBuffers()
{
    var triangleVertices = [
        -0.5, -0.5, 0.0,
        0.5, -0.5, 0.0,
        0.0, 0.0, 0.0,
    ];

    triangleVertexPositionBuffer = gl.createBuffer();
    gl.bindBuffer(gl.ARRAY_BUFFER, triangleVertexPositionBuffer);
    gl.bufferData(gl.ARRAY_BUFFER, new Float32Array(triangleVertices), gl.STATIC_DRAW);
}
```



triangleVertexPositionBuffer
isimli VBO'yu aktifleştir

Tek bir üçgenin köşe pozisyonları için VBO oluşturma

(-0.5,-0.5) , (0.5,-0.5) ve (0,0) köşe pozisyonlarına sahip bir üçgen çizdirmek için bu pozisyonları tutan bir VBO oluşturmamız gerekir.

```
function setupBuffers()
{
    var triangleVertices = [
        -0.5, -0.5, 0.0,
        0.5, -0.5, 0.0,
        0.0, 0.0, 0.0,
    ];

    triangleVertexPositionBuffer = gl.createBuffer();
    gl.bindBuffer(gl.ARRAY_BUFFER, triangleVertexPositionBuffer);
    gl.bufferData(gl.ARRAY_BUFFER, new Float32Array(triangleVertices), gl.STATIC_DRAW);
}
```

*triangleVertexPositionBuffer
isimli VBO'yu aktifleştir*


- `gl.ARRAY_BUFFER`: Buffer containing vertex attributes, such as vertex coordinates, texture coordinate data, or vertex color data.
- `gl.ELEMENT_ARRAY_BUFFER`: Buffer used for element indices.

Tek bir üçgenin köşe pozisyonları için VBO oluşturma

(-0.5,-0.5) , (0.5,-0.5) ve (0,0) köşe pozisyonlarına sahip bir üçgen çizdirmek için bu pozisyonları tutan bir VBO oluşturmamız.

```
function setupBuffers()
{
    var triangleVertices = [
        -0.5, -0.5, 0.0,
        0.5, -0.5, 0.0,
        0.0, 0.0, 0.0,
    ];

    triangleVertexPositionBuffer = gl.createBuffer();
    gl.bindBuffer(gl.ARRAY_BUFFER, triangleVertexPositionBuffer);
    gl.bufferData(gl.ARRAY_BUFFER, new Float32Array(triangleVertices), gl.STATIC_DRAW);
}
```



*triangleVertices verilerini aktif
VBO'ya 32 bit floating point
tipinde kopyala*

Tek bir üçgen çizdirme

```
function drawScene()
{
    vertexPositionAttribute = gl.getAttribLocation(glProgram, "aVertexPosition");
    gl.enableVertexAttribArray(vertexPositionAttribute);

    gl.bindBuffer(gl.ARRAY_BUFFER, triangleVertexPositionBuffer);
    gl.vertexAttribPointer(vertexPositionAttribute, 3, gl.FLOAT, false, 0, 0);
    gl.drawArrays(gl.TRIANGLES, 0, 3);
}
```


Tek bir üçgen çizdirme

```
function drawScene()
{
    vertexPositionAttribute = gl.getAttribLocation(glProgram, "aVertexPosition");
    gl.enableVertexAttribArray(vertexPositionAttribute);

    gl.bindBuffer(gl.ARRAY_BUFFER, triangleVertexPositionBuffer);
    gl.vertexAttribPointer(vertexPositionAttribute, 3, gl.FLOAT, false, 0, 0);
    gl.drawArrays(gl.TRIANGLES, 0, 3);
}
```

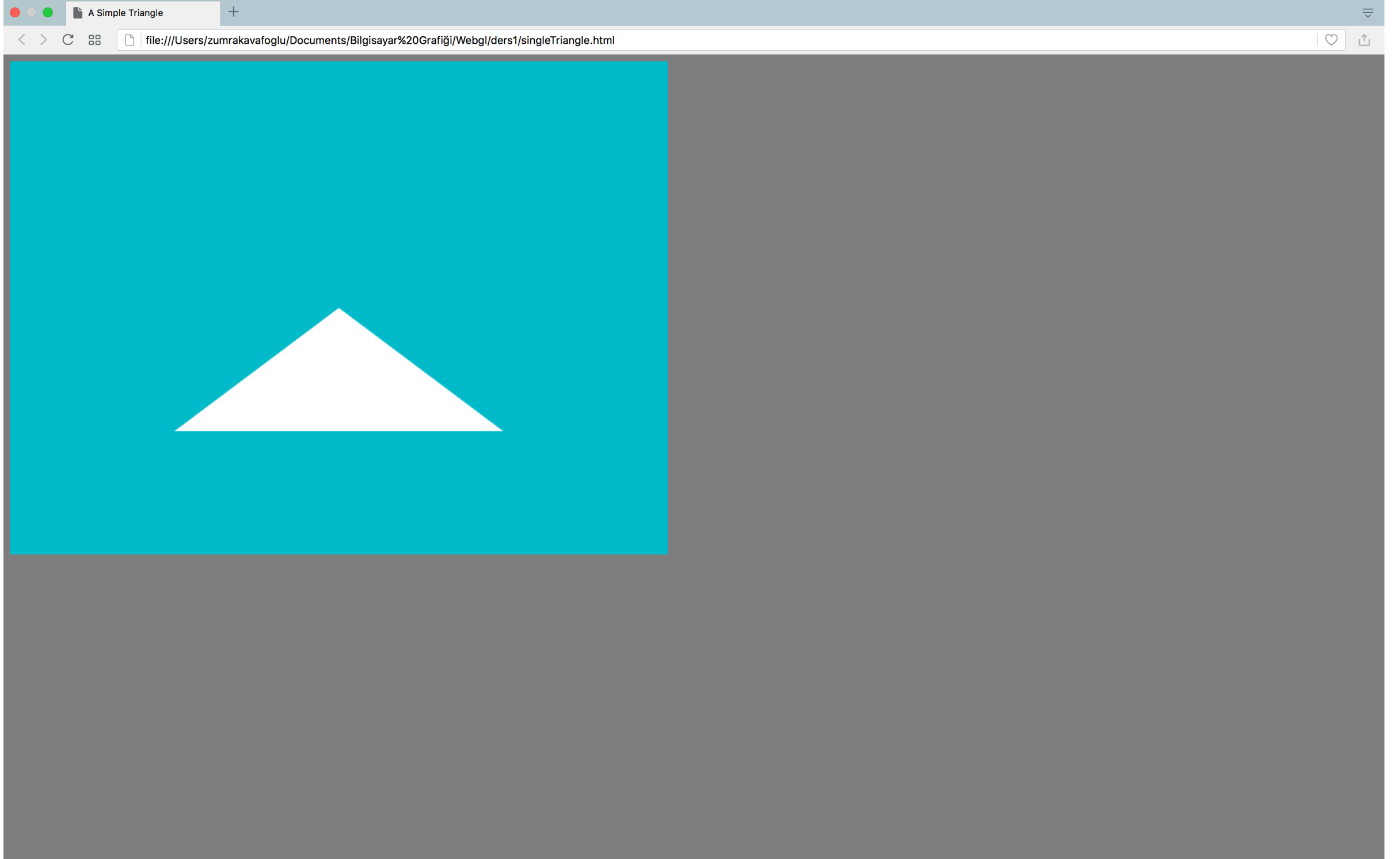
Primitif tipi



Özelliğin (köşe pozisyonu) sayısı

Özelliğin (köşe pozisyonunun) boyutu

Tek bir üçgen çizdirme



Alıştırma: Bir üçgen ve bir kare çizdirme

- Siz de aşağıdaki gibi bir üçgen ve bir kare çizdirmeyi deneyin.
- **İpucu:** Kare için yeni bir buffer oluşturun
- **İpucu:** Yalnızca dört köşelik bir dizi kullanarak çizdirmeye çalışın(TRIANGLES'dan farklı bir primitif tipi kullanarak bunu yapabilirsiniz.)

